EMA as a Physical Method for Extracting Secret Data from Mobile Phones

[1]Driss Aboulkassimi, [2]Jacques Fournier, [1]Laurent Freund, [2]Bruno Robisson and [2]Assia Tria

[1]Département "Systèmes et Architectures Sécurisés", Ecole Nationale Supérieure des Mines de Saint Etienne, France

[2]Laboratoire "Systèmes et Architectures Sécurisés", CEA-LETI Minatec, France

[1]lastname@emse.fr; [2]firstname.lastname@cea.fr

Abstract

Today's mobile phones have diverse functions and features such as calling, Internet surfing, game playing, banking, storage of personal and professional data. Given that these devices run an increasing amount of added value applications, the number of the software attacks on them has drastically increased. This study shows that the mobile platforms, especially their constituent components running security-related applications, could also be good targets for hardware attacks where sensitive data stored in the mobile phone are extracted using physical methods. This article discusses the feasibility and presents the results of a technique involving the extraction of secret keys by using the Electromagnetic Analysis (EMA) on software implementations of the Advanced Encryption Standard (AES) running on Java mobile phones. Obtaining these keys can be used for forensic purposes or to recover encrypted data that could have been enciphered using such keys.

Keywords

AES; Electromagnetic Attacks; Java Mobile Phones; Secret Keys.

Introduction

Until recently, security was not one of the priorities of mobile phone designers, except for some specific products principally intended to certain categories of business users and governments. Apart from the IMEI (International Mobile Equipment Identity) or SIM (Subscriber Identity Module) lock protections, the security was based principally on the smart card (SIM) for some features like network authentication, key derivations or for storing the mobile phone contacts of the user. With the advent of multi applicative smart phones, there is a growing need for security: for applications handling personal or professional data and information about the user's activities; for uses such as Internet shopping, game playing and banking. It has become crucial to ensure a consistent level of security in all the constituents of the mobile terminal in order to close all the possible attack routes.

The study presented in this article focuses on the analysis of hardware techniques that can be used to extract sensitive data from modern mobile phones. We target the secret key used in encryption algorithms like the AES that can be used for encrypting data or for network authentications. In this paper, we describe how a technique, called ElectroMagnetic Analysis (EMA), used on smart card like devices, can be adapted to a Java-based smart phone in order to extract information about the AES key used in some "off-the-shelf" implementations.

In this paper, we first provide a review of data extraction techniques that have already been presented for mobile devices. Then we present the AES and the Correlation Electro-Magnetic Analysis (CEMA) on AES. The mobile Java platform used in our experiments is discussed in order to explain in which conditions the analysis was done. Next, the experimental set-up, technical difficulties met and proposed solutions are explained. Spectral Density based Approach (SDA) and Template based Resynchronisation Approach (TRA), two approaches implemented to "remove" the misalignments obtained on the measurements, are described and compared. Two AES implementations are then analysed using CEMA and the results are presented. Finally countermeasures are discussed in order to protect software cryptographic libraries against such attacks.

Previous Research & Existing Techniques

Modern mobile phones like those called smart phones are multi applicative, multi function devices that increasingly manipulate and store personal and professional data. Unfortunately for user privacy and fortunately for forensic investigation, to extract these sensitive data, several invasive, semi-invasive and non-invasive physical techniques exist and can be adapted to wireless platforms.

An example of an invasive attack is the dump of the

Flash memory of the mobile device. This technique first consists in physically extracting the Flash memory (M. Breeuwsma, 2007), and then reading its content by using a Flash memory chip programmer. One of the disadvantages of this technique is that it can be destructive. This technique also requires a software driver and the right memory content interpretation (K. Kim, 2007). Neither is trivial when the data-sheets of the Flash memories are not available.

Flash memory dumps can also be done via the JTAG port (Joint Test Action Group, IEEE Std 1149.1, 2001). In this case it can be classified as a semi-invasive attack. The JTAG interface is a hardware module added to a device for testing analogue and digital integrated circuits. It can also be used for debugging software. In (M. Breeuwsma, 2005), the authors show that the JTAG port access can be exploited to extract data from the Flash memory of mobile devices. Related works are reported in (S. Willassen, 2005), where the authors managed to dump the Flash memory of an "old" mobile phone.

To test the efficiency of this technique on a smart phone, a commercial probe from Amontec (JTAGKEY) was chosen for its compatibility with the ARM11 processor of our targeted mobile device. The probe was connected to the JTAG port through a PCB (Printed Circuit Board) as illustrated in Fig. 1, that was designed specially to be adapted to the physical board's constraints of our mobile device.

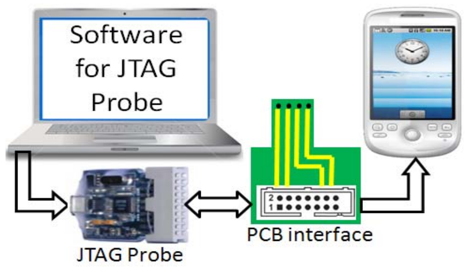


FIG. 1. JTAG PORT ACCESS ON MOBILE PHONE

Unfortunately, the "logical" connection could not be established between the probe and the processor. The performed manipulations allowed us to conclude that JTAG port access of the smart phone under test is protected (disabled) by the manufacturer. This shows that reading the phone's contents via the JTAG port is not as trivial as it was on older generations of mobile phones and that further investigations, which may lead to more invasive setups, would be needed to successfully implement this attack. To preserve the integrity of the mobile device that may be used in forensics proceedings, another physical technique for extracting secrets data was investigated.

This non-invasive technique is based on a category of attacks called "side channel attacks" (P.C. Kocher, 1996). This technique allows the extraction of sensitive data like the secret keys of encryption algorithms (for example the Advanced Encryption Standard - AES). It exploits the fact that some physical attributes such as the power consumption, the electromagnetic radiation or the duration of computation of the chip depend on its internal computations. Successful secret key extractions using the electromagnetic channel (a technique called Electro-Magnetic Analysis or EMA) have been reported in (K. Gandolfi, 2001) on smart cards and in (C. Gebotys, 2005) on a Java-based PDA (Personal Digital Assistant). To our best knowledge, this approach has not yet been tested on modern mobile phones and in this respect, we studied the efficiency of an EMA on a software AES running on the high-speed processor of a smart phone. The success of our attack is reported in section III. Given that on such platforms the AES can be used for server authentication protocols (like in PSK TLS) or to encrypt confidential information, the recovery of such secret keys can be useful for forensic and data recovery purposes, but can also put the user's privacy at stake if performed by a malevolent attacker.

Description of the AES and Principle of Electromagnetic Analysis

Side channel analysis, as a non-invasive method for extracting data like secret encryption keys, exploits the fact that some physical values such as the power consumption, the electromagnetic radiation or the duration of computation of the chip depends on its internal computations. It is of particular concern since it does not destroy the physical integrity of the circuit and it can be quickly mounted with cheap equipment. The main requirement for carrying such an analysis is that the attacker has to be able to measure several times (in practice, from hundreds to millions), the same cryptographic elementary operation with different operands. These measurements have to be done in the same conditions. In particular, the elementary operations have to be performed at the same time. In our analysis, we opted for the EM waves as physical characteristic to study because we could locally target the relevant chip. Using a characteristic like power consumption would have been tricky in the sense that there are too many sources of parasitic noise on a complex system board like that of a mobile phone.

Description of the AES

The AES is an iterative algorithm that performs encryption on data blocks of 128 bits as input and output, using key sizes of 128, 192 or 256 bits respectively in 10, 12 or 14 rounds according to the size of the key. The algorithm includes two separate processes: one for the key scheduling to derive the round keys from the initial secret key and the second one for data encryption. Decryption is also divided into two separated processes: the first one is for the inverse key scheduling and the second one is for the data decryption. In this paper, key sizes of 128 bits are used and we consider the encryption scenario. The structure of the AES-128 (referring to the AES using keys of 128 bits) is illustrated in Fig. 2. Each round key Ki is derived iteratively from the previous one as described in (NIST, 2001).

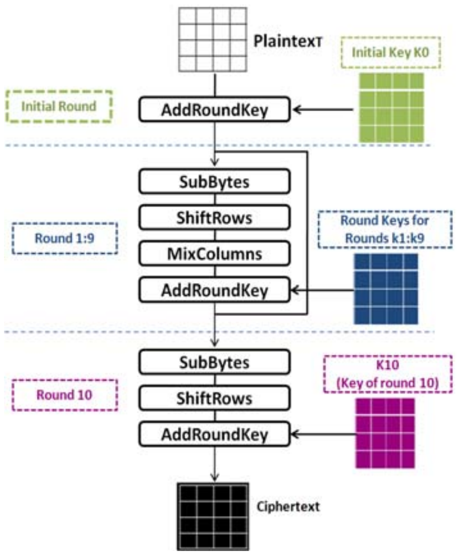


FIG. 2. AES ALGORITHM

AES Softwar Implementation

Lots of AES software implementations exist and, to develop a cryptographic Java Micro Edition (Java ME) application (since it is the system used on our device under test), developers have the choice among:

- Developing their own implementation according to the cryptographic algorithm specifications.
- Using a commercial or an open source implementation, e.g. the Bouncy Castle library .
- Using a manufacturer's implementation, e.g. the JSR 177.

The first approach requires development work without the help of any external commercial library. Consequently, neither a virus, nor a Trojan is possible. The second one has the advantage of using all the knowledge of an expert company or community. But royalties must sometimes be paid or open source license rules have to be followed. The last one seems to be the best solution in terms of performance and security because the manufacturer is responsible for the implementation. However the JSR 177 is today only available for very few mobile phones.

In this paper we only investigated about the first and the second scenarios, using a 32-bit processor as targeted hardware platform. In order to optimize the code size and the performances of our own AES implementation, we chose to combine the SubBytes and ShiftRows operations into one operation, implemented as a lookup table of 8 x 32 bits. In the Bouncy Castle library, the "fast algorithm" version of the AES has been chosen: three of the AES operations (SubBytes, ShiftRows and MixColumns) are grouped into one operation using a lookup table of 8 x 32 bits.

Principle of Correlation Electro Magnetic Analysis

The acquisition set-up is illustrated in Fig. 3. It consists of an EM probe, an oscilloscope, the trigger mechanism described below and a PC. The acquisition of the EM curves is performed as follows.

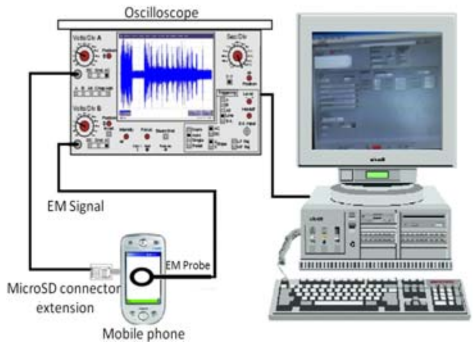


FIG. 3. EXPERIMENTAL SET-UP

The mobile phone executes, in a stand-alone mode, a large number (typically hundreds) of AES encryptions. At the beginning of each encryption, a signal is triggered as explained in a later section. This signal launches the oscilloscope which acquires the EM curves measured through the EM probe.

A "sleep time" is introduced between two encryptions. This sleep time provides the necessary time for the oscilloscope to send the EM curves to the PC that stores them.

Correlation Electromagnetic Analysis (CEMA) is an EMA technique based on the statistical analysis of the correlation between the EM waves measured during an AES computation and the data processed (secret key). An approach to CEMA would consist in the following steps:

- **Step1.** Measure the EM emanation. Let d be a set of D different plain texts for encryption d = (d_1 ,...,d_D). To each di is associated an EM curve t_i = ($t_{i,1}$,..., $t_{i,T}$) of T points each.
- **Step1b.** (optional) Modify mathematically the EM curves to obtain a set of t'_i = ($t'_{i,1}$,..., $t'_{i,T'}$) of T′ points.
- **Step2.** Choose intermediate results of the executed algorithm f(d, k) with k being a small part of the key (also called "guessed key"). Let K be the set of all the possible values of k.
- **Step3**. Compute some intermediate values $v_{i,j}$ = f(d_i, k_j) for i = 1, …,D and j = 1, …,K.
- **Step4.** Correlate the intermediate values with the EM waves. Hamming-distance or Hamming-weight models (E. Brier, 2004) consumption models may be used.
- **Step5.** Compare the measured EM values T (or the modified ones T′) with these theoretical EM values. The result is a matrix R of size K x T.

The line index of the highest values of the R is the index of the key actually used (correct key guess). The straightforward place for an attacker to find the secret key is during the first round (DPA Book, 2007). Therefore, the intermediate values used in Step3 are computed according to the SubBytes outputs' Hamming weights.

Setting up the EM Acquisition Bench

Unlike EMA on "classical" targets such as smart cards, we have faced two major difficulties. The first one is the choice of the experimental set-up that is slightly different from those classically described in the literature for smart cards. The second one is due to the complexity of the software running on the smart phone. As the AES is running inside a Java Virtual Machine (JVM), which is a complex software, the elementary operations of the AES were not always executed at the same time (i.e. they were not synchronized). The main requirement for EMA was not thus readily met. In order to overcome this difficulty, two techniques have been tested and compared. Before performing CEMA on a mobile phone, several points have to be first considered for the experimental set-up:

- Choice of the AES software implementation.
- Generation of a trigger signal to synchronize the software on the mobile phone with our acquisition platform.
- Choice of the EM probe.
- Physically accessing the phone's chip inside the mobile phone.
- Software for automatic acquisitions of EM traces.

The first point has been discussed previously. Concerning the forth point, the battery has been removed and the power supply was provided through wire connections. We shall now detail the three remaining points.

Trigger Signal

In order to signal to the oscilloscope that the EM wave acquisition can begin, a trigger signal, synchronized with the software running on the smart phone, has to be generated. This signal must be sent just before starting the AES encryption. Since all Java ME applications are executed in a sand box, we cannot have direct input/output accesses during the AES execution itself. The external inputs/outputs on a mobile phone are the screen, the loudspeaker and connections like http/SMS/Bluetooth and file access. The best solution we found that guaranteed an acceptable response delay was to access to a file stored on an external micro-SD card.

To facilitate the access to the micro-SD's PINS (Fig. 4), a special extension for the micro-SD's connector has been manufactured Our tests showed that in order to obtain a trigger signal, a simple open file instruction (Algorithm in Fig. 5) can be used to activate the PIN 2 of the micro-SD interface thus providing a deterministic trigger signal.

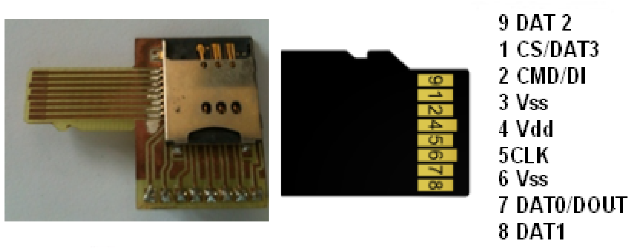


FIG. 4. MICRO SD CARD PINOUTS

```
public void trigger() {
    try {
            FileConnection file;
            file=Connector.open(file:///SDCard/myfile.txt);
            fclose(file); }  }
```

FIG. 5. ALGORITHM FOR MICRO SD TRIGGER

Choice of EM Probe

The choice of the probe is crucial to perform a successful analysis based on EM measurements (E. De Mulder, 2010). Some EM probes used to perform CEMA on smart card like devices are described by Mounier et al. in (B. Mounier, 2012). We first considered using some of the latter probes. However, as such probes (with diameters ranging from 70μm to 250μm) were too small with respect to the phone's die size (≈1cm^2), we had problems finding a proper position for the probe and making sure that we were measuring the relevant EM waves. We rapidly concluded that such probes were not appropriate for performing EM analysis on the phone's chip. From there we focused on probes large enough to "cover" the phone's chip. (C.K.Kim, 2008) describes the successful implementation of an EMA on an FPGA executing the ARIA algorithm (which is a fast symmetric key algorithm derived from the AES) using a "commercial" probe which has a diameter of 25mm and a bandwidth of 30MHz to 3GHz. For setting up our analysis on the phone's chip, we chose the same commercial probe equipped with a pre amplifier.

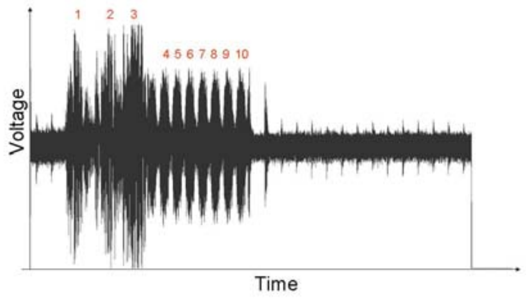


FIG. 6. EM CURVES CAPTURED BY HOMEMADE PROBE

Measurements were made on the phone's chip during the execution of our software AES using the chosen commercial probe. As shown in Fig. 7, the execution of the ten rounds of the AES could clearly be seen with a high signal to noise ratio.

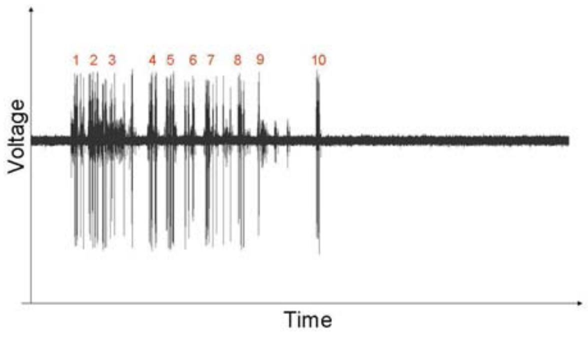


FIG. 7. EM CURVES CAPTURED BY COMMERCIAL PROBE

In order to illustrate the efficiency, and added value, of this commercial probe in our set-up, we performed the same measurements but this time using a home made probe (a solenoid coil of ten loops of diameter 1cm). A measurement made with the homemade probe is illustrated in Fig. 6 showing that a lower signal to noise ratio was obtained compared to when the commercial probe was used.

Automatic Acquisition Sequence

Table I sums up the characteristics and parameters of the equipment and conditions used during the acquisition phase. The 16 bytes of the text to encrypt are chosen by randomly changing only 1 byte, according to the attacked secret key byte. Other bytes are kept fixed in order to reduce measurement noise.

TABLE I ACQUISITION CONDITIONS AND EQUIPMENTS CHARACTERISTICS

Equipment	Characteristics
Target Circuit	RISC Processor, 32bits Clock frequency: 370Mhz
Oscilloscope	Lecroy 3Ghz, resolution 200 s/div Sampling frequency 1GSamples/s
EM Probe	30MHz-30GHz bandwidth
PC	Xeon 2.67 Ghz, RAM 12Go
Soft scope Driving	Labview interface Ethernet connection with oscilloscope
Plain texts to Encrypt	2000 random plain text changing only one of 16 the bytes

Analyzing the measured EM Curves: Handling the Misalignments

The following initial observations highlight the difficulty of interpreting the electromagnetic field of the execution of a Java program on a mobile phone. Several explanations can be derived from the Fig. 8 that describes the workflow from editing a Java program to executing it on the processor. Java is a compiled language.

Compilation techniques do a static analysis of the source code in order to reduce the execution time. Moreover, Java is intended to let application developers "write once, run anywhere". For that purpose, the Java Virtual Machine (JVM) interprets the Java byte code, issued from the compilation step. The portability is performed by the availability of a JVM for classical computer architectures. In parallel to the Java interpreter process, the execution of the just-in-time (JIT) compiler process raises concerns described later. The haphazard phenomena that appear on mobile phones are usually not present in the case of the more deterministic Java (smart) cards.

First, the "Garbage Collector" has been identified as one of these phenomena. The garbage collector consists in automatically cleaning the memory. The problem is that this process is launched in an uncontrolled way from a user point of view. Therefore, some EM curves are not exploitable (like in Fig. 9) because the signal is indistinguishable from noise. As the garbage collection consumes power and thus increases the electromagnetic radiation of the chip, the curves where the "Garbage Collector" is triggered are discriminated simply by computing their temporal average. The average of such curves appears to be 20% higher than those curves without garbage collector.

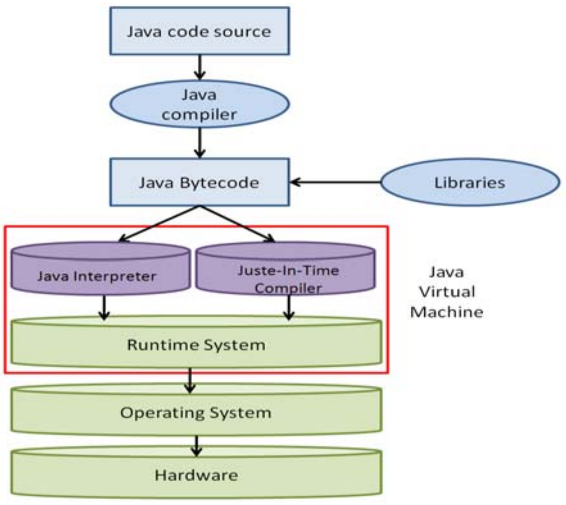


FIG. 8. JAVA PROGRAMME EXECUTION

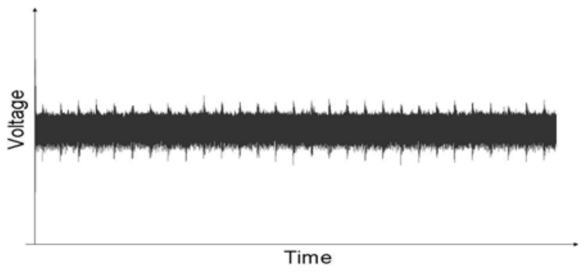


FIG. 9. CURVE ACQUIRED DURING GARBAGE COLLECTION: NO AES VISIBLE

Second, the "Just-In-Time-Compiler" was identified as a possible problem. Indeed, the "Just In Time Compiler" is an optimization executed by the virtual machine to dynamically speed-up the multiple execution of a set of instructions. It consists in compiling the instructions on the fly during their first execution. This phenomenon is visible in Fig. 7 where the first round is longer than the following ones.

These two issues introduce EM curves' misalignments. The temporal shift (Fig. 10) among EM curves varies between 0 and 110 ms and may impact the CEMA.

Third, the Java operating system is multi-threaded, consequently daemon processes could introduce additional delays. In order to find a workaround to all the EM curves' misalignments issues, a timing synchronization methodology based on the detection of the beginning of the first round was investigated. The statistical analysis failed due to temporal shift between instructions within the round itself. To solve this problem two successful solutions are described namely SDA and TRA.

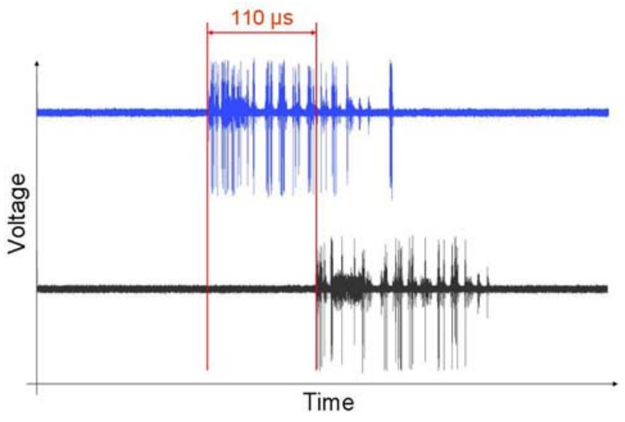


FIG. 10. MISALIGNMENT BETWEEN EM TRACES

EMA with Spectral Density Based Approach:

Such misalignments could also be due to random delays that could be introduced as a countermeasure in AES hardware and software implementations. Several techniques in signal processing exist and could be exploited, such as in (Jasper G, 2011) and (N. Homma, 2006). Another synchronization technique consists in performing the correlation not in the temporal domain but in the frequency domain. This technique is based on the fact that the Power Spectrum Density (PSD) of a "shifted signal" and the PSD of a "not shifted signal" are the same. In (C.C. Tiu, 2005), the authors show the efficiency of such an EM analysis on a very high speed embedded system. In (Zhang, P., 2009), the authors show that even if random delays are introduced, the CEMA attack in the frequency domain is still efficient on a program executed without a virtual machine on a processor running at less than 12 MHz. Unlike previous researches, our paper shows the efficiency of CEMA on a AES Java program executed on a virtual machine on a high speed circuit (400MHz). (O. Schimmel, 2010) exploits the simulation of the power consumption by using CPA (Correlation Power Analysis). Their analysis in the frequency domain allowed them to find the secret key.

We have performed on our own implementation of the AES, the EM attack in the frequency (or spectral) domain. It consists in computing the PSD of the EM curves in the optional **step 1b** described previously (with the Hamming Weight as the model for the EM signal). The result of the correlation is represented in Fig. 11 for one key-byte where the correlation factor of the correct key guess is distinguishable from the others. With this result, the efficiency of the CEMA attack has been shown on our AES J2ME implementation with resynchronisations performed in the spectral domain.

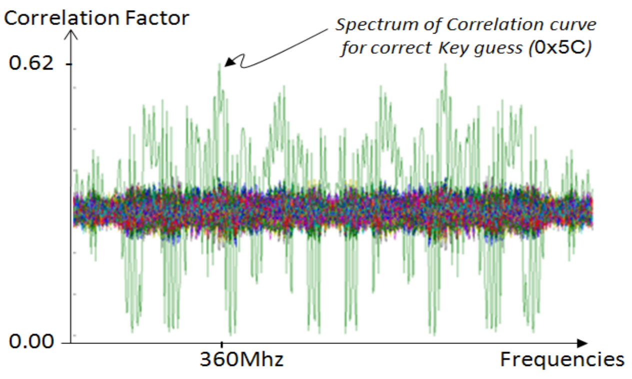


FIG. 11. CORRELATION RESULT FOR ONE KEY BYTE IN THE FREQUENCY DOMAIN

EMA Using Template Resynchronisation Approach:

During our experiments, we observed that the EM signature corresponding to the SubBytes parts of our AES implementation were very similar. It can be explained by the fact that the SubBytes are implemented as a LUT (Look-Up-Table), i.e. an access to an array, and that this access emits a characteristic EM signature. We use this property in order to resynchronize our curves.

The proposed approach consists first in developing a piece of code of a LUT access. Its signature is then measured and defined as the reference pattern (or “template”) (Fig. 12).

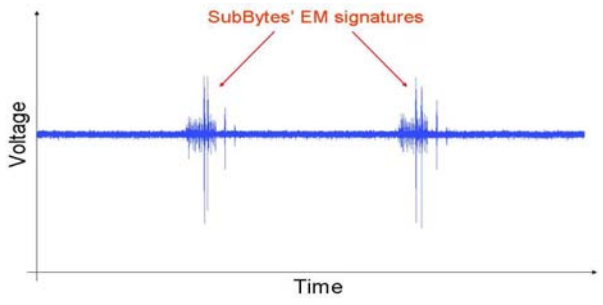


FIG. 12. ZOOM ON SUBBYTES SIGNATURES

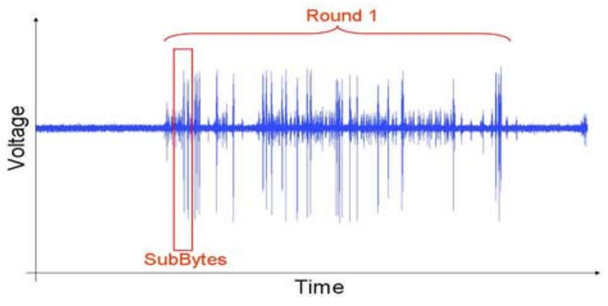


FIG. 13. IDENTIFICATION OF EM SIGNATURE OF LUT ACCESS

Then, the optional **step 1b** described previously has consisted in extracting the pieces of curves similar to such a template in the whole set of the raw EM curves (as represented in Fig. 13).

The sliding window technique correlation has been used to extract these pieces of curves. Thanks to this technique, each byte of the key used by our own implementation of the AES has been recovered with only 256 EM curves (Fig. 14). The correlation factor is about 72%. Using more curves increases the contrast between the correlation curve for the correct guess compared to the other 255 wrong guesses (Fig. 14).

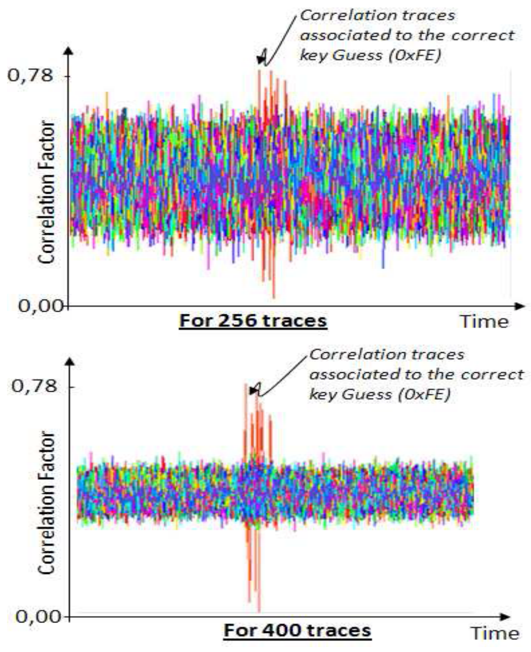


FIG. 14. CORRELATION RESULT FOR ONE KEY BYTE

Comparison between SDA and TRA

With the previously described results, it has been proven that SDA and TRA are efficient techniques to manage desynchronisation introduced by a mobile phone’s JVM during an EMA. The differences between both methodologies are summarized in Table II. As shown in this table, the SDA needs to select curves without “Garbage Collector” while the selection is automatically realized by the SubBytes operation identification technique of the TRA. An important point to note is that the first one requires four times more curves than the second one. On the opposite, the second one needs to build a template (i.e. the signature of a particular operation) before performing the attack. The real disadvantage in the first one is the number of samples (millions) compared to the second one (hundreds). Consequently more computer resources are necessary to perform the DSP and to apply the CEMA.

TABLE II : COMPARISON BETWEEN THE TWO PROPOSED APPROACHES

	SDA	TRA
Number of curves	20800	4096
Sorting curves	yes	no
Number of samples/curve	1 Million	600
Computing time	28h	1h

We showed that CEMA on our AES implementation on a mobile phone is also possible despite “irregularities” introduced by the JVM. By comparing both proposed approaches, the fastest has been identified. The next interesting challenge was to attack a ‘commercially used’ AES implementation using TRA.

EMA on Bouncy Castle’s AES

Bouncy Castle is an open source lightweight library used by Java applications that need cryptography. It supports standards like Transport Layer Security, Public Key Infrastructure and Certificate Management Protocol. In our experiments, we targeted the AES library in Bouncy Castle. To begin, a first acquisition was made. Visually, only 6 “patterns” were seen, which at first seemed in contradiction with our implementation (Fig. 15) where the 10 AES rounds could be seen. The challenge was to find the key without any knowledge of the source code. We made the hypothesis that the implementation of the SubBytes could be the same as on a 32-bit processor architecture, i.e. an 8x32 LUT. Therefore the template previously defined for our implementation, when using the TRA method, could be used for the identification of the Bouncy Castle’s SubBytes function operation.

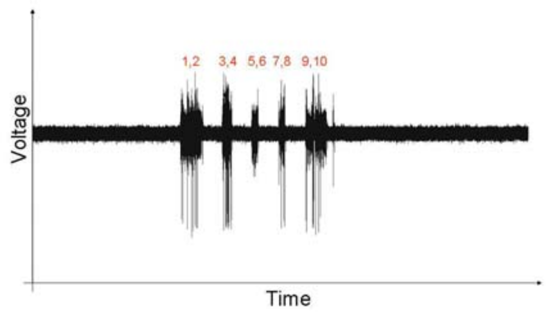


FIG. 15. EM CURVE OF BOUNCY CASTLE’S AES

Processings have been made in this manner and it has been discovered that the CEMA succeeds with only 250 curves (for each byte of key) with a correlation factor of 77% (Fig. 16).

This study demonstrates that an EM attack is possible without knowing the source code of an external library. This kind of attack could be dangerous because it could be generalized to all Java AES implementations on a 32-bit processor where a 8 x 32 LUT is used to implement the SubBytes operation.

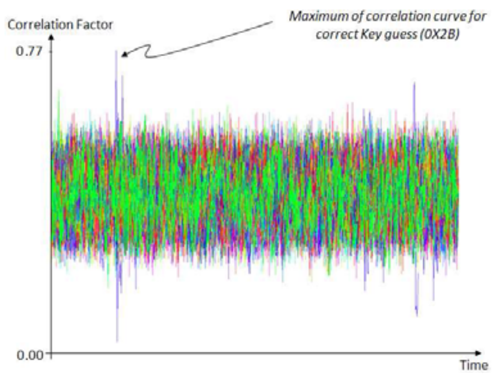


FIG. 16. CORRELATION RESULT FOR BOUNCY CASTLE AES (FOR ONE KEY BYTE)

Discussion on Countermeasures

CEMA like attacks, which we have adapted to mobile phones, have been implemented on devices like smart cards for some time now. In the later field, countermeasures against such attacks have been thoroughly studied. In this section we provide an overview of such countermeasures that mobile phone manufacturers could integrate to protect cryptographic calculations against CEMA like attacks. Among these techniques we distinguish among countermeasures implemented on the hardware level, the software level, in the packaging and on the applicative side:

- **Hardware countermeasures**: Those are features defined at the technological or architectural levels of the chip design to leverage the impact of the attacks on the chip itself. In the design phases of the mobile phone chip, the manufacturer may integrate hardware secure IP blocks like those proposed in (M. Agoyan, 2011). Another approach could be to adopt technologies like multi-rail encoding (R. Soares, 2008). Since CEMA needs synchronised EM traces, it is also possible to use random noise generator blocks (D. Page, 2003), that create EM perturbations and generate noise.
- **Software countermeasures**: This approach is based on the modification of the ciphering algorithm in the low level “driver” libraries or in the higher-level application software layers. Data masking (M-L. Akkar, 2001) is one of these software techniques where the input data and the

sensitive keys handled are "randomised" by adding a random value. Like in the hardware case, software random delays can be added within the AES software implementation (D.Naccache, 2005).

- **Packaging countermeasures**: Those are security features that can be added either in the way the chips are embodied into their package or the way the whole phone is packaged at the PCB board level. Such techniques are commonly used in point of sales terminals, (PCI/PIN, 2011). For example, sensors can be integrated to detect any illegal opening of the device's encasing and thus limit the physical access to the telephone's chips.
- **Applicative countermeasures**: Those are tweaks that can be added in the way applications use the sensitive routines so that attackers do not have a total control over the attack(s) he is carrying, thus reducing the chance for the attack to succeed. Concerning smart phones, hardware and packaging countermeasures may have a too high impact on their cost and their development cycle unless for cases where performance is important, in which case having dedicated hardware accelerators might be cost efficient.

Conclusion

In this paper we present what is, to our best knowledge, a first published study on physical attacks against modern mobile phones. Because such terminals now manipulate and store more and more private and confidential data, it has become vital to develop techniques to measure the resistance of such devices against physical attacks like those typically used in the smart card industry. Such techniques can either be used to measure to what extent a user's privacy is guaranteed or be used as a forensics method for unearthing encryption keys used to cipher data stored inside the phone. We first rapidly explored techniques like the use of JTAG probes without any success. We then set up a test bench and adequate analysis tools to successfully perform CEMA against the AES of the Bouncy Castle library. This shows that such libraries have to embed countermeasures against such attacks. Future work might consist in testing the technique against AES running on other smart phone's software platforms or against other cryptographic libraries like RSA or in setting up other powerful semi-invasive techniques like fault attacks.

ACKNOWLEDGMENT

The experiments were done on the MicroPackS platform and funded by the Secure Communications Solutions (SCS) cluster's FUI AAP10, project Calisson2.

REFERENCES

B. Mounier, A-L. Ribotta, J. Fournier, M. Agoyan and A. Tria, EM probes characterisation for security analysis, in 'Cryptography and Security: From Theory to Applications', pp 248-264, LNCS 6805, March 2012.

C.C. Tiu, A new frequency-based side channel attack for embedded Systems. MS thesis, Dept. of Electrical and Computer Eng., Univ. of Waterloo, 2005.

C. Gebotys,S. Ho and A. Tiu. EM Analysis of Rijndael and ECC on a PDA, Technical Report:CACR 2005-13, Dept of Electrical and Computer Engineering, 2005.

C.K. Kim, M. Schlaffer and S.J. Moon, Differential Side Channel Analysis Attacks on FPGA Implementations of ARIA, in ETRI Journal, vol. 30, num. 2, April 2008.

D. Naccache, P. Q. Nguyen, M. Tunstall and C. Whelan. Experimenting with faults, lattices and the DSA. In the proceedings of PKC 2005, LNCS 3386, pages 16-28, 2005.

D. Page. Defending against cache based side channel attacks. Information Security Technical Report, 8(1):3044, 2003.

E. Brier, C. Clavier and F. Olivier. Correlation power analysis with a leakage model. In Proceedings of CHES 2004, LNCS 3156, August 2004.

E. De Mulder, Electromagnetic Techniques and Probes for Side- Channel Analysis on Cryptographic Devices, PhD thesis, Arenberg Doctoral School of Science, Engineering & Technology, KUL, November 2010.

http://www.amontec.com/jtagkey-tiny.shtml, Last accessed, January 2013

IEEE Std 1149.1-2001. "IEEE standard test access port and boundaryscan architecture description", http://standards.ieee.org/reading/ieee/stdpublic/description/testtech/1149.1-2001 desc.html; July 23, 2001.

Jasper van Woudenberg, M. Witteman and B. Bakker. Improving diferential power analysis by elastic alignment. In the proceedings of CT-RSA 2011, pp 104-109, 2011.

jcp.org/aboutJava/communityprocess/final/jsr177/index.html, last accessed, January 2013

K. Gandolfi, C. Mourtel and F. Olivier, Electromagnetic analysis: concrete result, in the proceedings of CHES 2001, LNCS 2162 , pp. 251-261, Springer-Verlag, 2001.

K. Kim, D. Hong, K. Chung, and Ryou, "Data Acquisition from Cell Phone using Logical Approach", Proceedings of World Academy of Science, Engineering and Technology. Vol. 26. December 2007.

M. Agoyan, S. Bouquet, J. Fournier, B. Robisson, A. Tria, J-M. Dutertre and J-B. Rigaud, Design and characterisation of an AES chip embedding countermeasures, in IJIEI, Vol. 1, Nos. 3/4, 2011.

M. Breeuwsma, M. de Jongh, C. Klaver, R. Van der Knijff and M. Roeloffs, "Forensic Data Recovery from Flash Memory", Small Scale Digital Device Forensic Journal, Vol. 1, No. 1, 2007.

M. Breeuwsma, Forensic imaging of embedded systems using JTAG (boundary-scan), Digital Investigation, vol. 3, ed. 1, March 2006

M-L. Akkar and C. Giraud. An implementation of DES and AES, secure against some attacks, in the proceedings of CHES 2001, LNCS 2162, pp.309-318, 2001.

National Institute of Standards and Technology (NIST), Announcing the advanced encryption standard (AES), FIPS Publication, vol. 197, 2001.

N. Homma, S. Nagashima, Y. Imai, T. Aoki, and A. Satoh, Highresolution side-channel attack using phase-based waveform matching. CHES 2006, LNCS .4249, pp.187-200.

O. Schimmel, P. Duplys, E. Boehl, J. Hayek, R. Bosch, and W. Rosenstiel. Correlation power analysis in frequency domain. In proceedings of COSADE 2010.

P.C. Kocher, Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems, in proceedings of Crypto'96, LNCS 1109, pp. 104-113, 1996.

PCI security standards council Payment Card Industry / PIN TransactionSecurity / Point of Interaction / Modular Security Requirements, version 3.1, October 2011.

R. Soares, N. Calazans, V. Lomné, P. Maurine, L. Torres and M. Robert. Evaluating the robustness of secure triple track logic through prototyping, in the Proceedings of SBCCI'08, ACM, pp.193-198, 2008.

S. Mangard, E. Oswald and T. Poop, DPA Book, April 2007

S. Willassen. "Forensic analysis of mobile phone internal memory". In IFIP Int. Conf. Digital Forensics, pp 191-204 (2005).

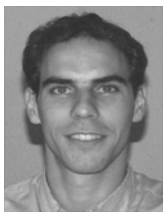
www.bouncycastle.org, last accessed, January 2013

Zhang, P., Deng, G., Zhao, Q., and Chen, K. EM Frequency Domain Correlation Analysis on Cipher Chips. In Proceedings of the 2009 First IEEE international Conference on information Science and Engineering.

Driss Aboulkassimi (S'09) received his engineering degree from the University of Montpellier. In 2010 he joined the Secure Architectures and Systems (SAS) lab, a joint team between the CEA-Leti and the Ecole Nationale Supérieure des Mines de St Etienne (ENSMSE. His research interests are security aspects of embedded systems.

Jacques Fournier worked 8 years for smart card manufacturer Gemalto before joining the CEA's SAS team in 2009 to work on smart card security, HW cryptographic accelerators and secure mobile systems. He graduated from SUPELEC, holds an MSECE from Georgia Tech and has a PhD from the Uni. of Cambridge.

Laurent Freund has been a lecturer in computer science at the ENSMSE for 14 years. He joined the SAS team 5 years ago. His research interests are mobile programming and secure mobile & embedded systems. He holds a Masters degree and a PhD from the Uni. of Evry.


Bruno Robisson studied electrical engineering at the Ecole Normale Supérieure de Cachan and received his PhD from University Paris VI in 2001. Since then, he has been a researcher at the CEA-Leti and joined the SAS team in 2005.. His main research topics are side channel attacks and development of secure cryptographic hardware.

Assia Tria is Research Director Habilited (HDR) by the French Ministry of Research. She received her PhD from University of Montpellier. She joined Gemalto in 1996 as a product engineer and then as a chip security expert. In 2005 she moved to the CEA-Leti to manage the SAS laboratory.